

# Formal Verification of Safety Properties for Ownership Authentication Transfer Protocol

Swaraj Bhat, Pradeep B.H, Keerthi S.Shetty and Sanjay Singh\*

August 22, 2012

## Abstract

In ubiquitous computing devices, users tend to store some valuable information in their device. Even though the device can be borrowed by the other user temporarily, it is not safe for any user to borrow or lend the device as it may cause private data of the user to be public. To safeguard the user data and also to preserve user privacy we propose and model the technique of ownership authentication transfer. The user who is willing to sell the device has to transfer the ownership of the device under sale. Once the device is sold and the ownership has been transferred, the old owner will not be able to use that device at any cost. Either of the users will not be able to use the device if the process of ownership has not been carried out properly. This also takes care of the scenario when the device has been stolen or lost, avoiding the impersonation attack. The aim of this paper is to model basic process of proposed ownership authentication transfer protocol and check its safety properties by representing it using CSP and model checking approach. For model checking we have used a symbolic model checker tool called NuSMV. The safety properties of ownership transfer protocol has been modeled in terms of CTL specification and it is observed that the system satisfies all the protocol constraint and is safe to be deployed.

## 1 Introduction

A ubiquitous computing (Ubicomp) or pervasive computing environment is imagined as a system with numerous invisible computers, sensors and actuators interacting with the user devices such as PDAs, Laptops, Mobile Phones etc. Data about the individuals who are a part of the ubiquitous environment is constantly being generated, transmitted, manipulated and stored. The user data present in the environment (in device or servers) is very sensitive. Protecting private data of every user in the environment is a major concern. Also

---

\*Sanjay Singh is with the Department of Information and Communication Technology, Manipal Institute of Technology, Manipal University, Manipal-576104, INDIA, E-mail: sanjay.singh@manipal.edu

in the this era of the mobile environment the user owns more than one portable devices like the PDAs, Laptops, Mobile Phones etc. with varying computing capabilities in order to access the variety of services that are being provided by the service providers. At times the user may tend to sell the device he owns. Since the device consists of the valuable information of the user and also will have the access to the valuable information present at the server, care should be taken to delete the information of the previous owner and store the details of the new owner in the device as well as the server.

Paulo Tam and Jan Newmarch [1] in their work have suggested the concept of transferring the ownership of the device. The owner (old owner) of the device will send the message to the device itself that it is being bought by the other user (new owner). The device will send the message to the new owner saying that its ownership is about to change to you (new user), do you accept or reject. The new owner sends the response to the device and the object will in turn send an acknowledgment on the status of the transfer to the old owner. However when the owner of the device is selling the device to the new owner, sending the message to the device itself does not seem feasible. Moreover to which device of the user, the device under sale is sending the message is not known. It is feasible if the new owner of the device has one more device under his ownership. But if the user has no other device previously and it is his first device then there is no possibility for the device under sale to send the message to its new owner asking his consent on the ownership transfer. In ubiquitous environment the ownership transfer has to be informed to the central server instead of informing to the device under sale.

Jurgen Bohn [2] has mentioned that the user can borrow or lend the device to his friend or the stranger. The data of a particular user can be retrieved from the personalization server at any time and from anywhere for a specific time. Once the time limit is exceeded, the session will expire and the user needs to quit the session or restart it. After using the device, the user can release the device and return it back to the owner of the device. But the very basic idea of sharing the personal device with a friend or a stranger may cause information to be public. This could be due to the other user being malicious (intentionally causing harm) by installing some kind of software which can record the data of the user or simply careless (unintentionally installing malicious software which can access the users data). Due attention should be paid to the fact that the device could come with old data, if the transfer is incomplete due to technical reasons such as network congestion or lack of connectivity. The owner of the device may also turn out to be malicious with respect to the other user. The user may install a software that records all the data that has been retrieved and sent from that device before encryption and after decryption. Later the user may be subjected to the impersonation attack. Moreover when the time limit is exceeded, there are chances that the user may have to end the session or restart it due to network latencies or unresponsive server when the user is trying to retrieve or release the data.

Yongming Jin et al [3] has described the transfer of RFID from the old owner to the new owner. They define a protocol to safeguard the privacy of the re-

spective owners by putting the clean stop between the transactions of the old and the new owners by means of a secret. The authors have suggested the use of RFIDs for the ownership transfer. But there are many security concerns with respect to the RFID tags. One of the primary RFID security concern is the illicit tracking of RFID tags. The tags can be read by anyone in the world and if the person who read the tag is malicious can pose a risk by either impersonating the user or trying to manipulate the user data and use it for a wrong purpose. RFIDs working at a shorter range are vulnerable to skimming and eavesdropping. Even though certain RFID tags use cryptographic features, the cost and power requirements are very high when compared to the simpler RFID tags. Thus, the cost and power limitation has compelled some manufacturers to implement cryptographic tags using substantially weak encryption schemes, which are weak to resist the sophisticated attack. Moreover, the power available in the handheld devices is limited; these tags cannot be incorporated in the devices.

Abdullah M. Alaraj [4] in his paper suggest that the users have to go to some officially designated place for buying or selling the merchandise and to complete the process of ownership transfer. He also makes an assumption that the certain equipments are required for ownership transfer and tries to improve the fairness by including the transfer of money through the bank servers. However going to an officially designated place that deals with buying or selling of merchandise is suitable only to the goods like cars or for real estate. This scenario will not be suitable when applied to ubiquitous computing devices. The process of ownership transfer requires only a Central Key Server (CKS) and a device meant for sale. Submitting users bank details to the third party might be risky at the time of payment. Even though if the system provides the best servers for transaction and promotes the users to submit their bank details to the device in an office meant for buying and selling of the merchandise, the device or the system in that office might turn out to be malicious.

In this paper we have modeled a newly proposed ownership authentication transfer protocol [5][6] which overcomes the limitations of the existing ownership transfer protocol. To the best of our knowledge any form of work in the field of formalizing ownership authentication transfer in ubiquitous computing devices has been minimal. In this paper, we have described basic process of ownership authentication transfer and formalized the safety properties using CSP approach. The safety properties of the proposed protocol is verified using a symbolic model checking approach. We have used a tool called New Symbolic Model Verifier (NuSMV) [7]. It searches the entire possible state space and checks for the correctness of the various specifications.

The remaining paper is organized as follows. Section 2 explains the operation of the proposed ownership authentication transfer protocol. Section 3 discusses modeling of the proposed protocol using CSP approach. Section 4 and 5 briefly discuss the concept of model checking and model checker tool used. Section 6 explains the modeling about of the safety properties in NuSMV. Section 7 discusses about the model checking results obtained for the safety properties for the proposed protocol. Finally, a conclusion has been drawn in

section 8.

## 2 Device Ownership Authentication Transfer Protocol

In this paper we propose a secure and fair protocol for ownership transfer of the ubiquitous computing devices. The user who is buying an old device from the other user has to undergo this process in order to successfully acquire the ownership of the device and start using it in the ubiquitous environment.

**Assumption:** The value or the price of the device has been agreed upon between the users before transferring the ownership of the device.

**Requisite:** The users should be in physical proximity and the whole process has to be carried out in the device which is under sale.

Table 1: Notations Used

Symbol	Meaning	Symbol	Meaning
$E_{CKS}$	Encryption Using Public key of CKS	CKS	Central Key Server
$N_A$	nonce generated by A	$N_B$	Nonce generated by B
$ID_A$	User ID or User Name of the user A	$ID_B$	User ID or User Name of the user B
$P_{CKS}$	Public key of the CKS	Ack	Acknowledgment
$PW_A$	Password of the User A	TempID	Temporary ID or Pseudo ID
OTC	Ownership Transfer Confirmation	OTR	Ownership Transfer Request

The previously existing user should introduce the new owner of the device to the CKS, in other words user A must transfer the ownership authentication credentials to the new user B. Once the new owner is introduced, the CKS will delete the credentials of the previous owner and save the credentials of the new owner for the same device. Once the ownership rights has been transferred to the new user, the old user at any cost will not be able to use the device. If in case the whole process of ownership transfer as mentioned below is not completed, neither of the users will be able to use the device. This also takes care of the scenario that if a device is stolen, the thief cannot use the device. The proposed ownership transfer protocol for a given ubiquitous device is explained below.

1.  $U_A \rightarrow CKS : E_{P_{CKS}}(ID_A || PW_A || N_A || OTR)$

The user A (Old User) sends the message to the CKS. The message consists of the user A ID, password of the user A, nonce of the user A and Ownership Transfer Request (OTR). This message is encrypted using the

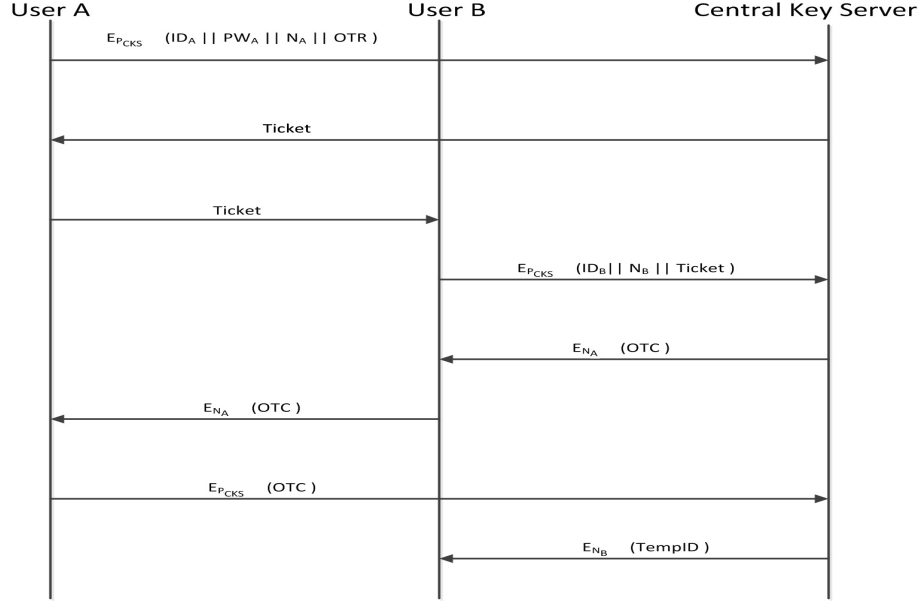


Figure 1: Diagram Showing Device Ownership Transfer Process

public key of the CKS. OTR consists of the ID of the user selling the device, ID and nonce of the user buying the device. OTR is also encrypted using the public key of the CKS, where  $OTR = E_{P_{CKS}}(ID_A || ID_B || N_B)$ . In this step the user A will introduce user B to the CKS.

2.  $CKS \rightarrow U_A : Ticket$

In response to the user A's request for ownership transfer, the CKS sends a ticket to the user A. The Ticket consists of the acknowledgment for ownership transfer to the user B. The ticket is encrypted using the public key of the CKS.

3.  $U_B \rightarrow CKS : E_{P_{CKS}}(ID_B || Ticket || N_B)$

The user A will now hand over the device to the user B. Now the user B sends his credentials to the CKS. The user needs to send user ID, nonce and the ticket got by the user A. The ticket will be in the device itself.

4.  $CKS \rightarrow U_B : E_{N_A}(OTC)$

Once the CKS receives the credentials of User B, the CKS sends the Ownership Transfer Confirmation(OTC) to the user B by encrypting it using nonce of the user A. This message consists of the information about the money to be transferred and the account details of the destination account.

5.  $U_A \rightarrow CKS : E_{P_{CKS}}(OTC)$

The user B will hand over the device to the user A and the user A will

decrypt the message, read the acknowledgment and then he sends the acknowledgment back to the CKS by encrypting it using the public key of CKS. By sending the acknowledgment back to the CKS, he confirms the ownership transfer of the device. Signing a particular message twice is required to strike the fairness in the deal. There may be some chances where either of the users may turn to be malicious. This is done in order to obtain a confirmation from the user who is selling the device.

6.  $CKS \rightarrow U_B : E_{N_B}(TempID)$   
On receiving the message, CKS completes the ownership transfer of the device by sending the temp ID to the user B. The temp ID is encrypted using the nonce of the user B.

The above explained process of device ownership transfer is summarized in the Fig. 1.

### 3 Modeling Device Ownership Authentication Transfer Protocol using CSP Approach

Communicating Sequential Processes (CSP) [8][9] is a notation for describing systems of parallel agents that communicate by passing messages between them. Security protocols work through the interaction of a number of processes in parallel that send each other messages. The typical security protocol involves several agents (often two: an initiator and a responder) and perhaps a server that performs some service such as key generation, translation or certification.

The Yahalom Protocol [10] representation of Ownership Authentication Transfer Protocol is as follows.

M1  $A \rightarrow CKS : \{ID_A \cdot PW_A \cdot N_A \cdot OTR\}_{P_{CKS}}$

M2  $CKS \rightarrow A : Ticket$

M3  $B \rightarrow CKS : \{ID_B \cdot ticket \cdot N_B\}_{P_{CKS}}$

M4  $CKS \rightarrow B : \{OTC\}_{N_A}$

M5  $A \rightarrow CKS : \{OTC\}_{P_{CKS}}$

M6  $CKS \rightarrow B : \{TempID\}_{N_B}$

The basic process of Ownership Authentication Transfer Protocol involves two agents: old owner and new owner of the device. The CSP description of protocol is given by equation (1).

The key  $P_{CKS}$  is the key of CKS that shares with Old Owner and New Owner of the device. The message is encrypted using public key of CKS.

$$\begin{aligned}
& OldOwner(A, N_A) \\
& = env? B \\
& : NewOwner \\
& \rightarrow send \cdot A \cdot CKS \cdot \{ID_A \cdot PW_A \cdot N_A \cdot OTR\}_{P_{CKS}} \cdot m \\
& \rightarrow \\
& \quad \square \\
& \quad \begin{array}{l} N_A \in Nonce \\ B \in NewOwner \\ m \in message \end{array} \left( \begin{array}{l} receive \cdot CKS \cdot A \cdot Ticket \rightarrow \\ send \cdot A \cdot CKS \cdot m \cdot \{OTC\}_{P_{CKS}} \rightarrow \\ Session(A, B, P_{CKS}, N_A, N_B) \end{array} \right)
\end{aligned} \tag{1}$$

$$\begin{aligned}
& NewOwner(B, N_B) \\
& = \quad \square \\
& \quad \begin{array}{l} N_A \in Nonce \\ A \in OldOwner \end{array} \left( \begin{array}{l} send \cdot B \cdot CKS \cdot \{ID_B \cdot Ticket \cdot N_B\}_{P_{CKS}} \rightarrow \\ receive \cdot CKS \cdot B \cdot \{OTC\}_{N_A} \rightarrow \\ receive \cdot CKS \cdot B \cdot \{TempID\}_{N_B} \rightarrow \\ Session(B, A, P_{CKS}, N_A, N_B) \end{array} \right)
\end{aligned} \tag{2}$$

The  $env? B$   $NewOwner$  is a representation how the processes local environment might tell it to open a session with agent B, this is formally expressed by equation (2).

Then Yahalom protocol is described as combination of users and servers. The Yahalom Process is expressed as:

$$Yahalom = OldOwner | NewOwner | CentralKeyServer.$$

When Intruder is present in the environment, the process is expressed as:  
 $System = Yahalom | Intruder$

### 3.1 Safety Properties

When Intruder is present in the environment, safety properties will be defined by introducing additional information into protocol descriptions to enable a description of what is expected of the system at particular points during a run of the protocol. The user who is buying an old device from the other user has to take care of safety properties in order to successfully acquire the ownership of the device and start using it in the ubiquitous environment.

#### 3.1.1 Secrecy

The old owner sends his credentials and OTR to the CKS. This message is kept secret until ownership of the device is transferred to the authenticated new

owner. The message *Claim\_Secret* will be inserted at the end of the description of the protocol run by old owner. Intruder cannot obtain any details during a run of the protocol whenever its secrecy is claimed.

An event *Claim\_Secret · OldOwner · NewOwner* message is used. This says that the message is kept secret during the run of the protocol.

The CSP description of secrecy property is given by equation (3) and (4).

$$\begin{aligned}
& \text{OldOwner}(A, N_A) \\
& = \text{env? } B \\
& : \text{NewOwner} \\
& \rightarrow \text{send} \cdot A \cdot \text{CKS} \cdot \{ID_A \cdot PW_A \cdot N_A \cdot OTR\}_{P_{CKS}} \cdot m \\
& \rightarrow \\
& \quad \square \\
& \quad \begin{array}{l} N_A \in \text{Nonce} \\ B \in \text{NewOwner} \\ m \in \text{message} \end{array} \left( \begin{array}{l} \text{receive} \cdot \text{CKS} \cdot A \cdot \text{Ticket} \rightarrow \\ \text{send} \cdot A \cdot \text{CKS} \cdot m \cdot \{OTC\}_{P_{CKS}} \rightarrow \\ \text{signal} \cdot \text{Claim\_Secret} \cdot A \cdot B \cdot N_B \rightarrow \\ \text{Session}(A, B, P_{CKS}, N_A, N_B) \end{array} \right)
\end{aligned} \tag{3}$$

$$\begin{aligned}
& \text{NewOwner}(B, N_B) \\
& = \quad \square \\
& \quad \begin{array}{l} N_A \in \text{Nonce} \\ A \in \text{OldOwner} \end{array} \left( \begin{array}{l} \text{send} \cdot B \cdot \text{CKS} \cdot \{ID_B \cdot \text{Ticket} \cdot N_B\}_{P_{CKS}} \rightarrow \\ \text{receive} \cdot \text{CKS} \cdot B \cdot \{OTC\}_{N_A} \rightarrow \\ \text{receive} \cdot \text{CKS} \cdot B \cdot \{\text{TempID}\}_{N_B} \rightarrow \\ \text{Session}(B, A, P_{CKS}, N_A, N_B) \end{array} \right)
\end{aligned} \tag{4}$$

### 3.1.2 Authentication

The old owner who is selling the device, initiates ownership transfer process of the device. In order to formalize authentication properties, two events are introduced during run of the protocol.

- *Commit · NewOwner · OldOwner*  
This says that NewOwner has completed a protocol run apparently with Old Owner.
- *Running · OldOwner · NewOwner*  
This says that OldOwner is following a protocol run apparently with NewOwner.

The CSP description for authentication property of device ownership transfer is given by equation (5) and (6).



$$\begin{aligned}
& OldOwner(A, N_A) \\
& = env? B \\
& : NewOwner \\
& \rightarrow send \cdot A \cdot CKS \cdot \{ID_A \cdot PW_A \cdot N_A \cdot OTR\}_{P_{CKS}} \cdot m \\
& \rightarrow \\
& \quad \square \\
& \quad N_A \in Nonce \\
& \quad B \in NewOwner \\
& \quad m \in message \\
& \quad \left( \begin{array}{l} receive \cdot CKS \cdot A \cdot Ticket \rightarrow \\ signal \cdot Running\_OldOwner \cdot A \cdot B \cdot N_A \cdot N_B \rightarrow \\ send \cdot A \cdot CKS \cdot m \cdot \{OTC\}_{P_{CKS}} \rightarrow \\ Session(A, B, P_{CKS}, N_A, N_B) \end{array} \right)
\end{aligned} \tag{5}$$

$$\begin{aligned}
& NewOwner(B, N_B) \\
& = \square \\
& \quad N_A \in Nonce \\
& \quad A \in OldOwner \\
& \quad \left( \begin{array}{l} send \cdot B \cdot CKS \cdot \{ID_B \cdot Ticket \cdot N_B\}_{P_{CKS}} \rightarrow \\ receive \cdot CKS \cdot B \cdot \{OTC\}_{N_A} \rightarrow \\ receive \cdot CKS \cdot B \cdot \{TempID\}_{N_B} \rightarrow \\ signal \cdot Commit\_NewOwner \cdot A \cdot B \cdot N_A \cdot N_B \rightarrow \\ Session(B, A, P_{CKS}, N_A, N_B) \end{array} \right)
\end{aligned} \tag{6}$$

## 4 Model Checking

As systems to be designed become more and more complicated, it is not sufficient at all to check the correctness of designs only by simulation. Subtle design errors can easily survive even under intensive and massive simulation. Also, detecting design errors in the late design stages is extremely costly and must be avoided as much as possible.

Formal verification is to mathematically prove that the behavior allowed by given specification (properties) contains the behavior performed by designs. It is essentially an exhaustive check on every possible behavior of designs that is related to the given specification.

Model checking is an automatic method to prove such correctness and is now becoming to be widely used in real design environments. Model checking is basically an exhaustive search in all possible states in the designs by checking whether the given specification is satisfied in all of them. That is mostly an implicit exhaustive search on state space of designs in the sense that state space of designs are represented symbolically instead of individually. This is why state-of-the-art model checking programs can verify designs having up to  $10^{100}$  or more states [11].

Specification for model checking is a set of properties that the designs must satisfy. Property can be described either in temporal logic or automaton. Temporal logic is an extension to traditional logic with temporal operators by which

we can describe relationships among variables in different time frames.

## 5 NuSMV

NuSMV is a symbolic model verifier tool for the formal verification of finite state systems. NuSMV allows us to check finite state systems against specifications in the linear temporal logic and Computational Tree Logic (CTL) [12]. The input language of NuSMV is designed to allow the description of finite state systems that range from completely synchronous to completely asynchronous [7]. It provides a modular hierarchical description and reusable component. NuSMV is mostly used for describing the transitions of a finite Kripke Structure [13].

NuSMV works only with finite data types such as boolean, scalar and fixed array. The main features of NuSMV are its functionalities, architecture and quality of implementation. NuSMV allows analysis of specification expressed in CTL and Linear Temporal Logic (LTL) using Binary Decision Diagram (BDD) and SAT-based checking [12].

We have used NuSMV for modeling basic process of ownership authentication transfer protocol and verification of its safety properties.

## 6 Modeling Safety Properties in NuSMV

As per section 3, the secrecy and authentication properties can be viewed as model is shown in Fig.2 and Fig.3 respectively.

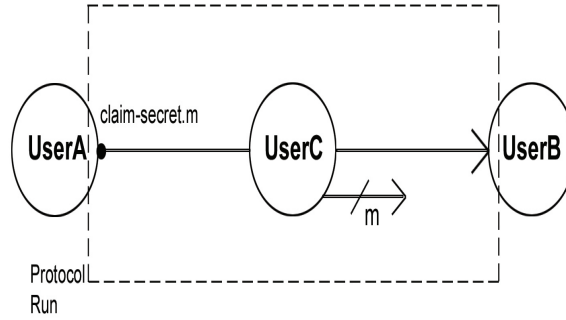


Figure 2: Model Showing a Secrecy Claim.

The various CTL specification for the safety properties can be written as:

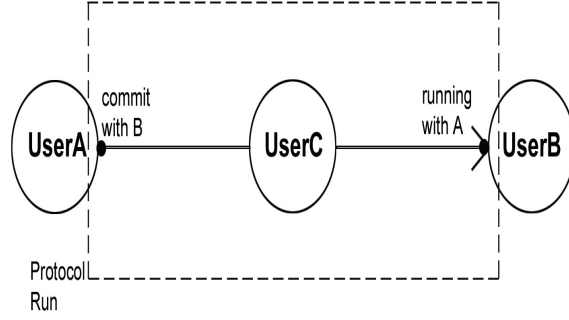


Figure 3: Model Showing an Authentication Claim.

- $AG(\text{OldOwner}(\text{claim\_secret})) \rightarrow AG(\text{NewOwner}(\text{message}))$   
This says that the UserA (OldOwner) initiates the ownership transfer process with UserB (NewOwner). The UserA claims for the secrecy of the message. The message is kept secret during run of the protocol. Intruder cannot access any kind of details of the session.
- $AG(\text{OldOwner}(\text{Running})) \rightarrow AG(\text{NewOwner}(\text{Commit}))$   
This says that the during the run of the protocol UserA is committed to UserB. UserB is following protocol run with UserA. Any kind of information is not revealed to the intruder present in the environment.

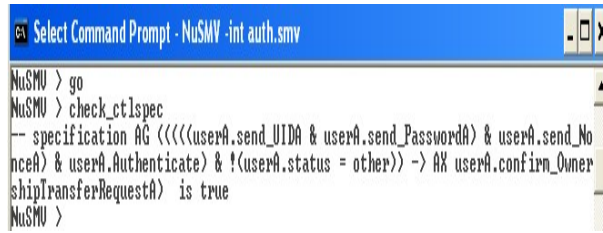
## 7 Model Checking Results and Discussion

Table 2: Specifications of the Protocol

Sl.No.	Case	Specification
1	UserA[1]=1 & UserA_claim_secret[1][2]=1	True
2	UserA[1]=1 & UserA_commit[1][2]=1	True
3	UserB[2]=1 & UserB_running[2][1]=1	True
4	UserC[2]=1 & UserC[2][1]=0	False
5	UserC[1]=1 & UserC[1][1]=0	False

Table 2 shows the various constraints imposed on the system and results of the corresponding specifications.

1. The first specification in the table depicts that UserA (Old Owner) initiates the ownership transfer process with UserB (New Owner), sends message to the CKS. The message consists of the UserA ID, Password, Nonce and OTR. The UserA claims for the secrecy of the message so that intruder present in the environment cannot access the details of the session. The secrecy property is verified through this specification. The specification simulation is shown in Fig.4.



```

Select Command Prompt - NuSMV -int auth.smv
NuSMV > go
NuSMV > check_ctlspec
-- specification AG <<<<(userA.send_UIDA & userA.send_PasswordA) & userA.send_NonceA & userA.Authenticate) & !(userA.status = other)) -> AX userA.confirm_OwnershipTransferRequestA) is true
NuSMV >

```

Figure 4: First Specification Verification Result Showing Result as True.

2. The second specification in the table shows the specification as true because the UserA(old user) introduce UserB(new user) to the CKS. UserB sends ID, Nonce and Ticket to the CKS. The UserA is committed with UserB. The specification simulation is shown in Fig.5.
3. The third specification in the table shows the specification as true because the UserA(old user) introduce UserB(new user) to the CKS. UserB sends ID, Nonce and Ticket to the CKS. The UserB is running with UserA. The specification simulation is shown in Fig.5.



```

Select C:\WINDOWS\system32\cmd.exe - NuSMV -int auth.smv
NuSMV > go
NuSMV > check_ctlspec
-- specification AG <<<<(userB.send_UIDB & userB.send_NonceB) & userB.send_Ticket) & userB.Authenticate) & !(userB.status = other)) -> AX userB.OwnershipTransferConfirmation) is true
NuSMV >

```

Figure 5: Second Specification Verification Result Showing Result as True.

4. The fourth specification says that UserC (Intruder) sends credentials of UserB to the CKS, trying to access the device. This is not possible. The specification returns false and counter example will be generated. The specification simulation is shown in Fig.6.
5. The fifth specification says that UserC (Intruder) sends UserA credentials to the CKS, trying to initiate the ownership transfer process of the device.

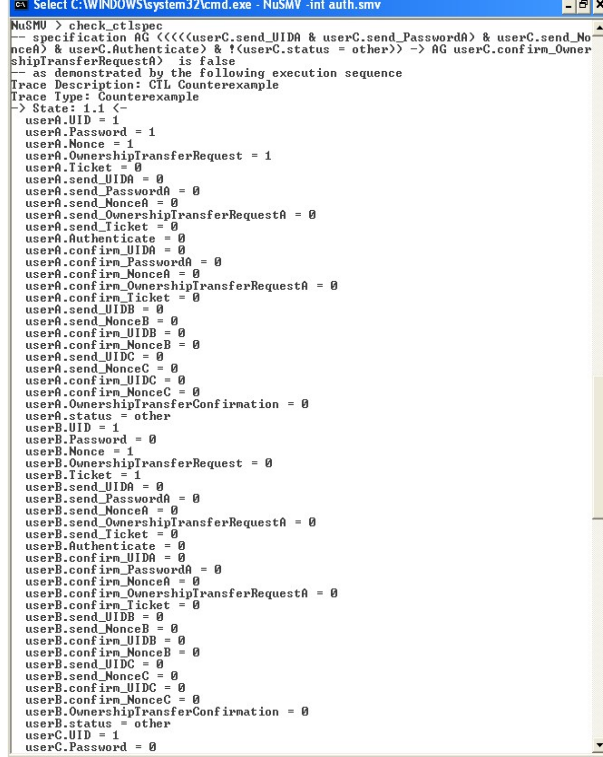
```

Select C:\WINDOWS\system32\cmd.exe - NuSMV -int auth.smv
NuSMV > check_ctlspec
-- specification AG (((userC.send_UIDB & userC.send_NonceB) & userC.Authenticat
e) & !(userC.status = other)) -> AG userC.OwnershipTransferConfirmation) is fal
se
-- as demonstrated by the following execution sequence
Trace Description: CTL Counterexample
Trace Type: Counterexample
-> State: 1.1 <-
  userA.UID = 1
  userA.Password = 1
  userA.Nonce = 1
  userA.OwnershipTransferRequest = 1
  userA.Ticket = 0
  userA.send_UIDA = 0
  userA.send_PasswordA = 0
  userA.send_NonceA = 0
  userA.send_OwnershipTransferRequestA = 0
  userA.send_Ticket = 0
  userA.Authenticate = 0
  userA.confirm_UIDA = 0
  userA.confirm_PasswordA = 0
  userA.confirm_NonceA = 0
  userA.confirm_OwnershipTransferRequestA = 0
  userA.confirm_Ticket = 0
  userA.send_UIDB = 0
  userA.send_NonceB = 0
  userA.confirm_UIDB = 0
  userA.confirm_NonceB = 0
  userA.send_UIDC = 0
  userA.send_NonceC = 0
  userA.confirm_UIDC = 0
  userA.confirm_NonceC = 0
  userA.OwnershipTransferConfirmation = 0
  userA.status = other
  userB.UID = 1
  userB.Password = 0
  userB.Nonce = 1
  userB.OwnershipTransferRequest = 0
  userB.Ticket = 1
  userB.send_UIDA = 0
  userB.send_PasswordA = 0
  userB.send_NonceA = 0
  userB.send_OwnershipTransferRequestA = 0
  userB.send_Ticket = 0
  userB.Authenticate = 0
  userB.confirm_UIDA = 0
  userB.confirm_PasswordA = 0
  userB.confirm_NonceA = 0
  userB.confirm_OwnershipTransferRequestA = 0
  userB.confirm_Ticket = 0
  userB.send_UIDB = 0
  userB.send_NonceB = 0
  userB.confirm_UIDB = 0
  userB.confirm_NonceB = 0
  userB.send_UIDC = 0
  userB.send_NonceC = 0
  userB.confirm_UIDC = 0
  userB.confirm_NonceC = 0
  userB.OwnershipTransferConfirmation = 0
  userB.status = other

```

Figure 6: Fourth Specification Verification Returns False and Gives a Counterexample.

The specification returns false and counter example will be generated. The specification simulation is shown in Fig.7.



```

c:\Select C:\WINDOWS\system32\cmd.exe - NuSMV -int auth.smv
NuSMV > check_ctlspec
-- specification AG <<<<(userC.send_UIDA & userC.send_PasswordA) & userC.send_No
nceA) & userC.Authenticate) & !(userC.status = other)>> -> AG userC.confirm_Owner
shipTransferRequestA) is false
-- as demonstrated by the following execution sequence
Trace Description: CTL Counterexample
Trace Type: Counterexample
-> State: 1.1 <-
  userA.UID = 1
  userA.Password = 1
  userA.Nonce = 1
  userA.OwnershipTransferRequest = 1
  userA.Ticket = 0
  userA.send_UIDA = 0
  userA.send_PasswordA = 0
  userA.send_NonceA = 0
  userA.send_OwnershipTransferRequestA = 0
  userA.send_Ticket = 0
  userA.Authenticate = 0
  userA.confirm_UIDA = 0
  userA.confirm_PasswordA = 0
  userA.confirm_NonceA = 0
  userA.confirm_OwnershipTransferRequestA = 0
  userA.confirm_Ticket = 0
  userA.send_UIDB = 0
  userA.send_NonceB = 0
  userA.confirm_UIDB = 0
  userA.confirm_NonceB = 0
  userA.send_UIDC = 0
  userA.send_NonceC = 0
  userA.confirm_UIDC = 0
  userA.confirm_NonceC = 0
  userA.OwnershipTransferConfirmation = 0
  userA.status = other
  userB.UID = 1
  userB.Password = 0
  userB.Nonce = 1
  userB.OwnershipTransferRequest = 0
  userB.Ticket = 1
  userB.send_UIDA = 0
  userB.send_PasswordA = 0
  userB.send_NonceA = 0
  userB.send_OwnershipTransferRequestA = 0
  userB.send_Ticket = 0
  userB.Authenticate = 0
  userB.confirm_UIDA = 0
  userB.confirm_PasswordA = 0
  userB.confirm_NonceA = 0
  userB.confirm_OwnershipTransferRequestA = 0
  userB.confirm_Ticket = 0
  userB.send_UIDB = 0
  userB.send_NonceB = 0
  userB.confirm_UIDB = 0
  userB.confirm_NonceB = 0
  userB.send_UIDC = 0
  userB.send_NonceC = 0
  userB.confirm_UIDC = 0
  userB.confirm_NonceC = 0
  userB.OwnershipTransferConfirmation = 0
  userC.UID = 1
  userC.Password = 0

```

Figure 7: Fifth Specification Verification Returns False and Gives a Counterexample.

## 8 Conclusion

In this paper, we have proposed a new ownership authentication transfer protocol for ubiquitous computing devices. The basic process of ownership authentication transfer and safety properties is described using CSP approach. We have used symbolic model checking approach to model the safety properties of the proposed protocol. The tool NuSMV helps us to verify the constraints imposed on the system by exploring the entire state space of the system. It provides a counter example along with the trace path to point to the location of error, if the system does not meet any of the constraints. The safety properties of a protocol is modeled efficiently using NuSMV. It is observed that all the constraints are met by the system developed and the proposed protocol is safe to be used in Ubiquitous environment.

## References

- [1] P. Tam and J. Newmarch, "Protocol for ownership of physical objects in ubiquitous computing environments," in *IADIS International conference E-Society*, 2004, pp. 614–621.
- [2] J. Bohn, "Instant personalization and temporary ownership of handheld devices," in *Mobile Computing Systems and Applications*, 2004. WMCSA 2004. Sixth IEEE Workshop on, dec. 2004, pp. 134 – 143.
- [3] Z. C. Yongming Jin, Huiping Sun, "Hash-based tag ownership transfer protocol against traceability," in *IEEE International Conference on e-Business Engineering- 2009*, 2009, pp. 487–492.
- [4] A. Alaraj, "Ownership transfer protocol," in *Internet Technology and Secured Transactions (ICITST)*, 2010 International Conference for, nov. 2010, pp. 1 –6.
- [5] B. Pradeep, "Privacy and Ownership Authentication of Ubiquitous Computing Devices," Master's thesis, Manipal Institute of Technology, May 2012.
- [6] B. Pradeep and S. Singh. (2012, Aug) Ownership authentication transfer protocol for ubiquitous computing devices. [Online]. Available: <http://arxiv.org/pdf/1208.1712.pdf>
- [7] A. Cimatti, E. M. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella, "NuSMV 2: An OpenSource Tool for Symbolic Model Checking," in *CAV '02: Proceedings of the 14th International Conference on Computer Aided Verification*. London, UK: Springer-Verlag, 2002, pp. 359–364.
- [8] P. Ryan, S. Schneider, M. Goldsmith, G. Lowe, and A. Roscoe, *The Modelling and Analysis of Security Protocols: the CSP Approach*. NY, USA: Pearson, 2010.
- [9] M. Pitt, "Modeling and verification of security protocols part i: Basics of cryptography and introduction to security protocols," Advanced Seminar Paper of Dresden University of Technology, 2002.
- [10] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *ACM Trans. Comput. Syst.*, vol. 8, no. 1, pp. 18–36, Feb. 1990. [Online]. Available: <http://doi.acm.org/10.1145/77648.77649>
- [11] J. R. Burch, E. M. Clarke, D. E. Long, K. L. McMillan, and D. L. Dill, "Symbolic model checking for sequential circuit verification," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 13, no. 4, pp. 401–424, 1994.

- [12] M. Huth and M. Ryan, *Logic in Computer Science: Modelling and Reasoning about Systems*. New York, NY, USA: Cambridge University Press, 2004.
- [13] (2012) Kripke structure. [Online]. Available: [http://en.wikipedia.org/wiki/Kripke\\_structure](http://en.wikipedia.org/wiki/Kripke_structure)